EP 501 Homework 2

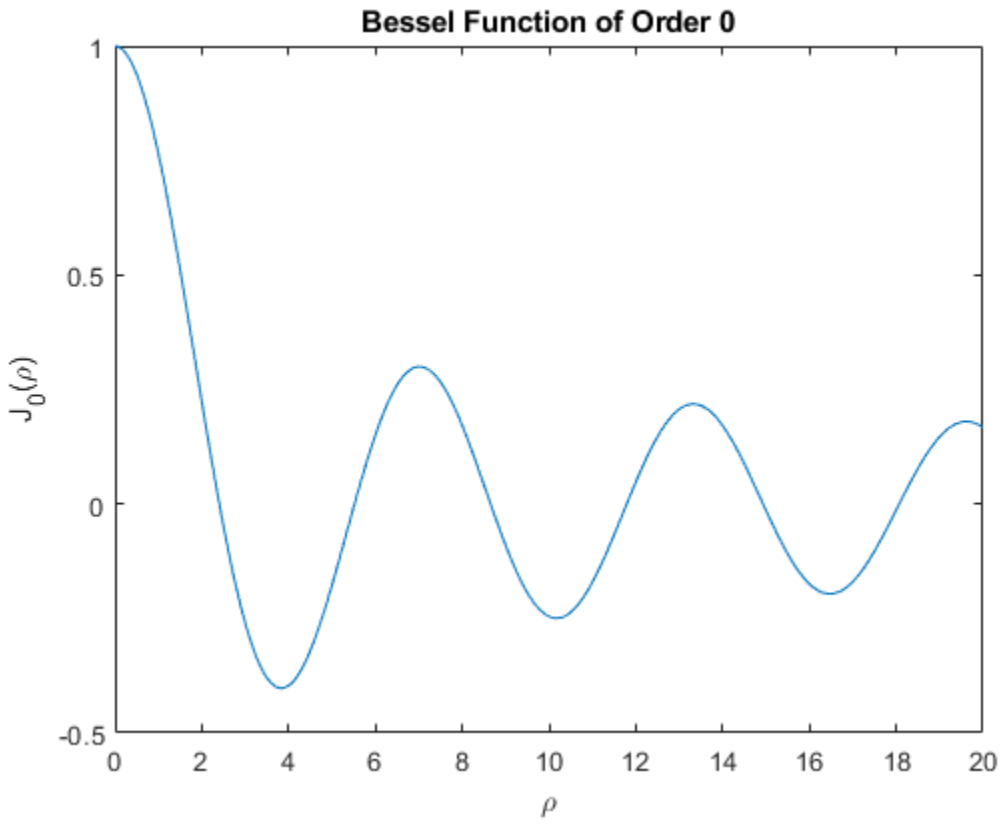
```
%Julio Guardado
clc; clear; close all;
```

Problem 1, Part B: first root of 0 order bessel function

```
%define function and setup
rho = linspace(0,20,200);
eta = .01;
maxit = 100;
tol = 1e-9;
f_rho = besselj(0,rho);

%plot function to find initial guess
figure(1);
plot(rho,f_rho)
xlabel('\rho');
ylabel('J_0(\rho)');
title('Bessel Function of Order 0')

%find first root using approximate newton method
x0 = 2.5;
[xNewtApprox, niter1,~] = newton_approx_bess(x0,eta,maxit,tol);
```



Problem 1, Part C: first six roots of order 0 bessel function

```
bess0_roots = zeros(1,7);       %allocate space for roots and number of iterations
niter = zeros(1,7);             %index 1 is only used for checking
x0 = 0;
j = 2;
while (j<8)                      %iterate using approx newton function
    x0 = x0 + 3;
    [bess0_roots(j),niter(j),~] = newton_approx_bess(x0,eta,maxit,tol);
    %check for convergence on the same root or out of order root
    if(bess0_roots(j) == bess0_roots(j-1))
        j = j-1;
        x0 = x0 + x0;
    else
        j = j+1;
    end

end

%display results
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp('Problem 1:')
disp('First six roots of 0th order Bessel function:')
disp(bess0_roots(2:end))
fprintf('\t(first 5 roots checked against Wolfram Mathworlds Bessel Function Zeros)\n')
fprintf('\t http://mathworld.wolfram.com/BesselFunctionZeros.html\n\n')
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Problem 1:
First six roots of 0th order Bessel function:
    2.4048    5.5201    8.6537   11.7915   14.9309   18.0711

	(first 5 roots checked against Wolfram Mathworlds Bessel Function Zeros)
	 http://mathworld.wolfram.com/BesselFunctionZeros.html
```

Problem 2.a

```
%get function handles
f = @(x) x.^5 - 15.*x.^4 + 85.*x.^3 - 225.*x.^2 + 274.*x -120;
fder = @(x) 5.*x.^4 - 60.*x.^3 + 255.*x.^2 - 450.*x + 274;

%define terms for root finding
maxit = 100;
tol = 1e-9;
n = 5;                      %order of poly
x0 = linspace(.75,4.75,5);
rootsa = zeros(1,n);

%find roots
for j = 1:n
    rootsa(j) = newton_exact(f,fder,x0(j),maxit,tol);
end
```

Problem 2.b

```
%get function handles
f = @(x) x.^3 - 3.*x.^2 + 4.*x - 2;
fder = @(x) 3.*x.^2 - 6.*x + 4;

%define terms for root finding
n = 3;                          %order of poly
x0 = [.5,.5+.5*1i,.5-1i];
rootsb = zeros(1,n);

for j = 1:n
    rootsb(j) = newton_exact(f,fder,x0(j),maxit,tol);
end

%display results
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp('Problem 2a:')
disp('Roots of x^5-15x^4+95x^3-225x^2+274x-120: ')
disp(rootsa)
disp('Problem 2b:')
disp('Roots of x^3-3x^2+4x-2: ')
disp(rootsb)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Problem 2a:
Roots of x^5-15x^4+95x^3-225x^2+274x-120:
    1.0000    2.0000    3.0000    4.0000    5.0000

Problem 2b:
Roots of x^3-3x^2+4x-2:
   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 - 1.0000i
```

Problem 3

```
%define coefficients of polynomial and function handles
coeffs = [1,-15,85,-225,274,-120];
coeffder = [5,-60,255,-450,274];

%find first root
x0 = .5;
n = length(coeffs);
roots = zeros(n-1,1);
roots(1) = newton_exact_poly(coeffs,coeffder,x0);   %use modified newton_exact that takes in
polynomial coefficients

%iterate polynomial division to get roots
poly = coeffs;
for i = 2:n-2
    %polydiv
    poly = polydiv(poly,roots(i-1));
    polyderiv = polyder(poly);

    if(i == n-2)
        break
    end

    %find new root
    x0 = x0+1;
    roots(i) = newton_exact_poly(poly,polyderiv,x0);
end

%find last two roots using quadratic equation
[roots(n-1),roots(n-2)] = quadrat(poly);

%Display results
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp('Problem 3:')
disp('Roots of Equation 1 using polynomial deflation:')
disp(roots)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Problem 3:
Roots of Equation 1 using polynomial deflation:
    1.0000
    2.0000
    3.0000
    4.0000
    5.0000
```

Problem 4a

```
%define functions
fm=@(x,y) x.^2 + y.^2 - 2.*x - y;
gm=@(x,y)(x.^2)./4 + (y.^2) - 1;
gradfm=@(x,y) deal(2.*x - 2,2.*y - 1);
gradgm=@(x,y) deal((1/2).*x, 2.*y);

%define terms for root finding
nx = 2; %order of x
ny = 2; %order of y

%initial guess
x0 = [3 -1];
y0 = [-1 2];
rootxa = zeros(1,nx);
rootya = zeros(1,ny);

%find roots
for i = 1:nx
    for j = 1:ny
        [rootxa(i),rootya(j),~,~] = newton2D_exact(fm,gradfm,gm,gradgm,x0(i),y0(j));
    end
end
```

Problem 4b

```
%define functions
fm2 =@(x,y,z) x.^2 + y.^2 + z.^2 - 6;
gm2 =@(x,y,z) x.^2 - y.^2 + 2.*z.^2 - 2;
hm2 =@(x,y,z) 2.*x.^2 + y.^2 - z.^2 - 3;

gradfm2 = @(x,y,z) deal(2.*x,2.*y,2.*z);
gradgm2 = @(x,y,z) deal(2.*x,-2.*y,4.*z);
gradhm2 = @(x,y,z) deal(2.*x,2.*y,-2.*z);

%define terms for root finding
nx = 2; %order of x
ny = 2; %order of y
nz = 2; %order of z

%initial guess
x0 = 3;
y0 = -1;
z0 = 2;

%find root
[rootxb,rootyb,rootzb,~,~] = newton3D_exact(fm2,gradfm2,gm2,gradgm2,hm2,gradhm2,x0,y0,z0);

%display results
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp('Problem 4a:')
```

```
disp('Roots of 2D system of equations:')
fprintf('\t1st root: (%f,%f)\n\n',rootxa(1),rootya(1))
fprintf('\t2st root: (%f,%f)\n\n',rootxa(2),rootya(2))

disp('Problem 4b:')
disp('One Root of 3D system of equations:')
fprintf('\t1st root: (%f,%f,%f)\n\n',rootxb(1),rootyb(1),rootzb(1))
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Problem 4a:
Roots of 2D system of equations:
        1st root: (2.000000,0.000000)

        2st root: (-0.000000,1.000000)

Problem 4b:
One Root of 3D system of equations:
        1st root: (0.999775,-1.732094,1.414320)
```

Published with MATLAB® R2018b